CVRPTW metaheuristic solutions

Liam Watson

November 3, 2022

1 Implementation

1.1 Problem Discussion and Definition

The Capacitated Vehicle Routeing Problem with Time Windows is defined as finding the shortest route for any n vehicles which cannot exceed a capacity (200) and must arrive within a customer's specified time window. As each customer is visited they have some "demand" which represents the capacity that will be used within the truck as well as a ready time and due time - the truck must arrive before the due date. However, in this study, we assume that if a truck arrived before the ready time it will wait until the customer is ready $time = readyTime_{customeri}$.

1.2 Genetic Algorithm (GA)

The first step in implementation was deciding on a representation for the CVRPTW that adequately represents the problem while still allowing the natural application of the genetic operations. For this representation, it was found that a two dimensional structure that can be flattened for genetic operation then brought back into two dimensions. The structure is displayed in Figure 1.

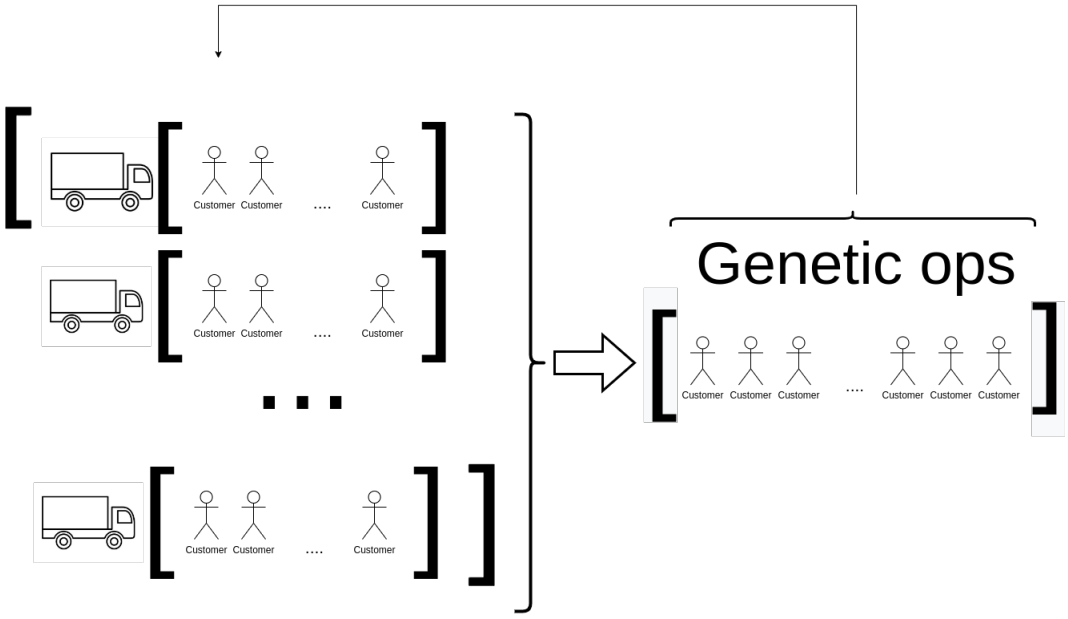

Figure 1: Showing the GA chromosome structure

The GA was implemented in java using swap mutation and order crossover (OX). Partially mapped cross-over (PMX) was attempted but proved inferior to OX likely due to the order crossover preserving local order better than PMX which is highly important to ordered crossover. For the introduction of additional genetic variation, it was found that swap mutation far outperformed inversion likely due to inversion failing to introduce long-distance changes as well as inversion's likelihood of breaking learned efficient time windows.

1.3 Ant Colony Optimisation (ACO)

The representation of the ACO routes is similar to those in the GA Figure 1, however, there is no need to flatten trucks - rather when each "ant" decides on a route separate decisions between n trucks each starting and ending at the depot. There are two points where we can encourage trucks to satisfy time windows and minimize distance - namely in the objective function and selection probability.

1.3.1 Objective function

$$f(r) = C_1 d_r + C_2 \Delta t_r \tag{1}$$

Where r is some ant's route, d_r is the distance traveled in the route, Δt_r is the total time that the ant is late by, C_1 and C_2 are constants to control the influence of each of the parameters (C_2 is set to a large value to encourage satisfying the required time windows as this term goes to zero when satisfied).

$$\Delta t_r = \sum_{i=0}^{n} time_{due} - time$$

Where n is the number of customers and depot's visited in an ants route.

1.3.2 Selection probability

In the typical selection probability, η represents the greedy tendency of an ant and is the distance from customer i to j. However, to encourage time windows to be satisfied we include a delta time punishment much like in the objective function 2.

$$\eta_{ij} = \begin{cases} 0 & \text{if } time \leq time_{due} \\ due_{time} - time & \text{otherwise} \end{cases} \tag{2}$$

1.4 Parameter Recommender

The parameter recommender implementation is done in python as it yields parallel execution more easily than in java. Separate implementations are constructed for the GA and ACO respectively where executions of the respective algorithm are done in parallel batches (size depending on hardware) to ensure an efficient search. Parameters are set for the respective algorithm via command line arguments.

2 Results

2.1 Genetic Algorithm

The best route found for the Solomon R101 data instance with 100 customers had a distance of VALUE. The route is given below - it should be noted that checks are included to ensure that trucks never exceed the maximum capacity, never arrive to customers late, and include all customers in the instance.

```
0       [ 0,32,31,67,33,71,0 ]
1       [ 0,64,65,50,47,49,0 ]
2       [ 0,22,74,88,58,14,0 ]
3       [ 0,30,35,36,25,26,0 ]
4       [ 0,28,70,52,21,2,0 ]
5       [ 0,96,99,17,39,44,0 ]
6       [ 0,40,24,68,57,5,0 ]
7       [ 0,53,89,9,85,18,0 ]
8       [ 0,77,80,4,69,81,0 ]
9       [ 0,60,93,6,100,7,0 ]
10      [ 0,62,86,38,92,101,0 ]
11      [ 0,46,37,48,87,98,0 ]
12      [ 0,73,3,16,13,78,0 ]
13      [ 0,54,95,97,61,94,0 ]
14      [ 0,43,15,45,41,27,0 ]
15      [ 0,63,12,91,11,90,0 ]
16      [ 0,84,83,20,8,19,0 ]
17      [ 0,29,66,72,10,51,0 ]
18      [ 0,76,23,42,75,59,0 ]
19      [ 0,34,82,79,56,55,0 ]
Fitness: 1782
Valid: true
TIME WINDOW PASSED
Length: 1782
```

2.2 Ant Colony Optimisation

The best result found for ACO on the Soloman R101 instance with 100 customers was VALUE with the same checks included in the GA.

```
    Best route: tour :
Truck 0: 0 53 26 54 24 80 0
Truck 1: 0 45 36 47 19 49 0
Truck 2: 0 40 87 97 37 100 0
Truck 3: 0 27 69 31 88 7 0
Truck 4: 0 28 12 76 50 1 0
Truck 5: 0 52 18 8 46 48 0
Truck 6: 0 15 57 43 91 93 0
Truck 7: 0 2 29 79 3 77 0
Truck 8: 0 92 59 99 85 86 0
Truck 9: 0 33 81 9 35 70 0
Truck 10: 0 95 98 61 84 17 0
Truck 11: 0 21 73 74 4 25 0
Truck 12: 0 30 51 20 66 32 0
Truck 13: 0 72 75 22 41 56 0
Truck 14: 0 6 94 96 13 89 0
Truck 15: 0 62 11 64 90 10 0
Truck 16: 0 42 14 44 16 38 0
Truck 17: 0 5 83 82 60 58 0
Truck 18: 0 39 23 67 55 68 0
Truck 19: 0 63 65 71 78 34 0
objectiveValue : 1694.898529291153
Delta Time: 0.0
Length: 1694.898529291153
```

With the following parameters:

```
    public final double startPheromoneValue = 0.0000005;
    public final int numberOfIterations = 150;

    public final int capacity = 200;
    public final int numberOfTrucks = 20;


    public final int numberOfCustomers = 100;

    public final int assignToTruck = numberOfCustomers/numberOfTrucks;

    //parameters
    public  double decayFactor = 0.5;
    public  double alpha = 1.5;
    public  double beta = 2;
    public  int numberOfAnts = 3500;

    public double timeWindowObjectivePunishment = 1000;
    public double timeWindowEtaPunishment = 1000;
    public double distanceObjectivePunishment = 1;

    public double distanceEtaPunishment = 0.5;

    public double Q = 1;
```

3 Parameter Recommender

The parameter ranges tested for the GA and ACO are included in tables X and Y respectively. The best parameters are displayed in Tables X and Y.

Parameter	From (inclusive)	To (inclusive)	step size
Cross Over ratio	0.5	0.7	0.05
Mutation Ratio	0.001	0.005	0.001
Population size	1000	3500	500

Table 1: Range of parameter values tested in GA

Parameter	From (inclusive)	To (inclusive)	step size
swarm size	10	3500	10,100,500,1500,3500
α	0.5	2.5	0.5
β	2	4	0.5
ρ	0.1	0.9	0.1

Table 2: Range of parameter values tested in ACO